Neural Time-Invariant Causal Discovery from Time Series Data

Saima Absar
Computer Engineering
University of Arkansas
Fayetteville, AR, USA
sa059@uark.edu

Yongkai Wu
Electrical and Computer Engineering
Clemson University
Clemson, South Carolina, USA
yongkaw@clemson.edu

Lu Zhang
Computer Engineering
University of Arkansas
Fayetteville, AR, USA
lz006@uark.edu

2023 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-8867-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IJCNN54540.2023.10192004

***Abstract*—Causal structure learning from observational data is an active field of research over the past decades. Although many approaches exist, such as constrained-based methods and score-based methods including the emerging deep learning-based methods, most of them address the static, non-dynamic setting. In this paper, we propose a score-based causal discovery algorithm named Neural Time-invariant Causal Discovery (NTiCD), which learns summary causal graphs from multivariate time series data based on the principle of Granger causality. NTiCD is a continuous optimization-based technique that leverages the power of deep neural networks to compute the score values. To this end, we use an LSTM to obtain the hidden non-linear representations of temporal variables in the time series data. Then, these features are aggregated using graph convolutional networks and decoded using an MLP that outputs the forecast of the future data values in the time series. The model is optimized based on a score function subject to regularized loss. The final output is a summary causal graph that captures the time-invariant causal relations within and between time series. We evaluate the performance of our algorithm on several synthetic and real datasets. The result analysis over a number of different datasets demonstrates the improvement in the accuracy of causal structure discovery of temporal data compared to other state-of-the-art methods.**

***Index Terms*—causal discovery, time series, summary causal graph, encoding-decoding framework, graph convolutional networks**

I. Introduction

Inferring the causal relations from observational data is a critical problem in many fields of science, economics, philosophy, etc. [8]. The traditional way of detecting causality involved carrying out controlled randomized experiments to introduce interventions that are often expensive, time-consuming, and in many cases, unethical or impossible [23]. With the widespread availability of digital data these days, it has become easier to extract causal information from the analysis of such data. Thus, data-driven approaches utilizing machine learning and artificial intelligence techniques have come to play a vital role in causal discovery. Existing causal discovery approaches can be categorized into two major classes, constraint-based and score-based methods. The first approach involves conditional independence testing between the variables to determine the causal direction according to some constraints [14]. The latter approach like [12] learns a Bayesian network by optimizing some predefined score function that assigns a score to each causal graph. Specifically, the deep learning extension to the score-based approach leverages the advantages of continuous optimization to minimize the score function (see survey [32]).

Although past studies of causality mostly involved non-temporal static data, many applications around us involve temporal data, for example, identifying causality in climate data [16], or analyzing influences among different regions of the brain using fMRI [27]. The causal discovery of time series involves identifying the underlying causal relationships among the variables in the temporal data. Models that can handle temporal data to infer causal relations are an important part of the field of causal discovery.

There has been increasing attention to causal discovery in time series data in recent decades [3], [8], [11]. One major principle of causal discovery in the temporal domain is Granger causality [9]. It states that a time series is a cause of another time series if the past values of the former can predict the future values of the latter, assuming that there are no hidden confounders. Based on that, both constraint-based and score-based approaches have been proposed. Deep learning methods for temporal causal discovery have also been studied in recent years [1], [17], [19], [20]. However, due to complex dynamics in time series, most methods are unable to capture the time-invariant causal relationship encoded in the data. They usually infer multiple window causal graphs with specific time lags instead of a single summary causal graph that captures the causal relations within and between time series without specifying the time information. In addition, these models are often limited to assumptions of linear equations.

In this paper, we propose a score-based causal discovery algorithm named Neural Time-invariant Causal Discovery (NTiCD). It discovers time-invariant causal structures as summary causal graphs for multivariate time series. Our method leverages the power of deep neural networks and continuous optimization. The architecture of NTiCD consists of three main components: encoding, aggregation, and decoding. The encoding component is a long short-term memory (LSTM) network that aims to obtain the hidden non-linear representations of temporal variables in the time series data. The ag-

This work was supported in part by NSF 1920920 and 2142725.

gregation component is a graph convolutional network (GCN) with the adjacency matrix of the causal graph as trainable parameters. It takes the representations from the encoding component as the input and outputs the aggregated information from the local neighborhood of each temporal variable. The decoding component is a multilayer perceptron (MLP) that predicts future data values in the time series based on aggregated encoded information. The loss function is regularized to control the magnitude of the adjacency matrix. Finally, we conduct end-to-end training to simultaneously optimize the parameters of all networks as well as the adjacency matrix of the causal graph.

Compared with existing approaches, our method yields a number of benefits. First, the output of NTiCD is a summary causal graph that does not assume stationarity, i.e., causal dependencies are repeated with the same time lag at all time points, or make any specific assumptions on the lag values. Second, NTiCD does not assume linear relations between time series and can be applied to non-linear equations. Third, the model does not assume that self-causes always exist and can infer self-causes for each temporal variable.

We conduct experiments using both synthetic and real-world datasets and compare NTiCD with several baseline methods. The synthetic data are generated according to predefined non-linear functions and random summary causal graphs. For real-world data, we use the Netsim dataset [27] which is made up of realistic simulated functional magnetic resonance imaging (fMRI) time series. We evaluate the performance of causal discovery based on multiple metrics. The results demonstrate an improvement in the performance of NTiCD over the state-of-the-art methods.

II. Related Work

The earliest approach for time series causal discovery is the Granger causality that was introduced almost half a century ago [9], [26]. Due to the limitation of its applicability to only linear processes, as addressed by Granger himself, several extensions have been proposed since then [17], [18], [29]–[31]. Many other methods for Granger causal discovery use vector autoregressive methods including [10], [13], [22]. VarLinGAM is a method proposed in [13] that estimates the structural vector autoregressive (SVAR) models by generalizing the linear non-Gaussian acyclic model (LiNGAM) [25]. The authors use the theory of non-Gaussianity to discover causality in temporal data, however, their method assumes linear models. TCDF [20] extends the theory of Granger causality to non-linear settings by using deep learning models. An attention-based convolutional network is adopted and the causal interpretation is made based on the kernel weights and attention score independently for each time series. However, these traditional Granger causality approaches generally suffer from scalability issues and do not perform well for a large number of variables.

Constraint-based algorithms can also be extended to time series data. Early constraint-based algorithms like PC and Fast Causal Inference (FCI) [28] build a causal graph using conditional independence testing with Markov condition and faithfulness assumption. Several algorithms have been developed based on PC and FCI, for example, PCMCI [24], LPCMCI [7], tsFCI [5], μ-PC [1], and one of the most recent FCITMI [2]. FCITMI addresses the problem of learning summary causal graphs by combining PC-like and FCI-like algorithms along with some entropy reduction principles. It uses a temporal mutual-information measure using a sliding-window technique. However, it is extremely slow for a large number of variables as it combines two constraint-based approaches that have large time complexities.

More recent approaches utilize deep learning models to extend the score-based method. DYNOTEARS [22] proposes a score-based technique to learn an SVAR model, which is also known as a dynamic Bayesian network (DBN), to infer a causal graph from time series. Although the network scales quite well with an increasing number of nodes, DYNOTEARS is based on a linear VAR model similar to FCITMI and VarLinGAM. A different framework for inferring multivariate Granger causality is proposed by [19] using Self-explaining Neural Networks (SENNs). Their model, referred to as GVAR, allows the detection of Granger-causal effect signs, i.e., applicable to both original and time-reversed data. These approaches decompose the temporal causal relations into different slices where each slice represents the causal relation with a specific time lag. Different from these approaches, our method analysis the causal relations between time series as a whole and directly outputs a summary causal graph.

Other related approaches include Amortized Causal Discovery (ACD) [17], which learns causal relations from data as different graphs but with shared dynamics using an encoder-decoder module. A similar approach is proposed in NSM [32] for both video and time-series data. Minimum Predictive Information Regularization (MPIR) [33] is another method that is based on minimizing a mutual information objective between each pair of time series given other time series. A similar information-theoretic approach is proposed by [4], which is a greedy-based algorithm that detects causality by data compression.

III. Method

Consider a dataset $\mathbf{X}$ consisting of d time series of equal length n, such that each time series i is a sequence represented as $X^i := X^i_{0:(n-1)} := \{x^i_0, x^i_1, \ldots, x^i_{(n-1)}\}$. For simplicity, we consider the case of a one-dimensional sequence, i.e., at each time step t $x^i_t \in X^i$ is a scalar. However, our method can be directly applied to multidimensional sequence cases, as will be shown in the experiments. A summary causal graph is a directed graph where each node in the graph represents a time series. An edge pointing from one node to another represents that the history of the former time series causes the future values of the latter time series in any form and with any lag. If an edge points from one node to itself (i.e., a self-loop), it means that this time series has a self-cause. We do not assume that self-causes always exist in every time series. Fig. 1 shows an example of a summary causal graph that consists of three nodes: X^1, X^2, and X^3, each of which represents a

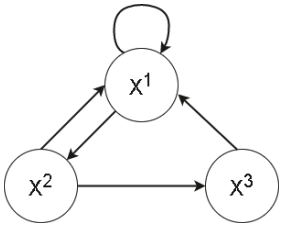


Fig. 1: An example of summary causal graph.

time series. Here, an arrow from X^1 to X^2 indicates that X^1 causes X^2. One can find in the graph the cycles [e.g., $X^1 \to X^2 \to X^3$], the self-loops [$X^1 \to X^1$], the colliders[$X^2 \to X^1 \leftarrow X^3$], and the confounders [$X^1 \leftarrow X^2 \to X^3$]. The goal of causal discovery is to learn a summary causal graph G from the data that captures the causal relationship among time series. We leverage the Granger causality for this purpose.

A. Granger Causality

Granger causality is one of the most basic principles in inferring causal relationships among observational time series data. It is based on the assumption that causes must precede their effects in terms of time steps. According to the Granger causality, if a time series X^1 'Granger' causes time series X^2 then the past of X^1 will contain statistically significant information to facilitate the prediction of future values of X^2. The classic Granger causality is mathematically formulated using linear regression models of stochastic processes [9]. For example, let X^2 be defined by the bivariate linear autoregressive model as follows:

$$x_t^2 = \sum_{j<t} \alpha(X_{0:j}^1, X_{0:j}^2) + \epsilon_t$$

where α is a linear autoregressive function and ϵ_t represents independent noises. Then, X^1 Granger causes X^2 if the statistical hypothesis test $\sum_{j<t} \alpha(X_{0:j}^1, X_{0:j}^2)$ is equivalent to $\sum_{j<t} \alpha(X_{0:j}^2)$ in predicting x_t^2 is rejected.

The major limitation of the classic Granger causality is that it is not applicable to dynamic non-linear cases. Next, we introduce the Neural Time-invariant Causal Discovery (NTiCD) framework that uses deep neural networks to replace the linear autoregressive function and leverages continuous optimization to discover Granger causality and the summary causal graph.

B. Overview of NTiCD

NTiCD is an encoding-decoding framework for learning the summary causal graph as a $d \times d$ adjacency matrix A from the time series data. Inspired by recent research in static causal discovery (e.g., [6], [15], [21], [34]–[36]), NTiCD consists of three components. The first component is an encoding module, which aims to learn a non-linear hidden representation to capture the historical information in the time series until time t. The second component is an aggregation module that aggregates information from the local neighborhood according to the adjacency matrix A. The third component is a decoding module that utilizes the aggregated historical information to make predictions of future values in the time series. All three components will be updated simultaneously in the training process according to a regularized loss that measures the accuracy of the prediction and the magnitude of the adjacency matrix A. Finally, matrix A is converted to the directed graph based on a certain thresholding technique. The rationale of NTiCD is that when A is closer to the true causal graph, more relevant information will be aggregated by the aggregation module and used by the decoding module for making predictions. Meanwhile, when the encoding and decoding modules are updated, they can better capture the significant information for the prediction. After the training converges, it is expected that A becomes close to a summary causal graph that demonstrates Granger causality.

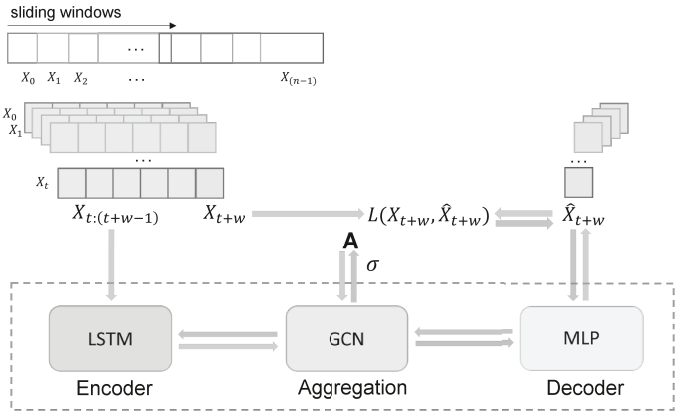


Fig. 2: An overview of the architecture of NTiCD.

An overview of NTiCD is demonstrated in Fig. 2 which explains how the input time series is fed to NTiCD to optimize the loss function and predict the causal relationship in terms of a summary causal graph via the three components. Below we explain each component in detail.

C. Architecture of NTiCD

Encoding. In this module, the historical information in the time series is extracted as non-linear representations. We leverage a deep neural network to learn what information should be encoded so that we do not make assumptions about the time lag of the causal dependencies or linear relations. It is worth noting that when selecting the network architecture, the network must be powerful enough to capture significant information, yet it cannot be too powerful and memorizes the whole sequence of data. For this purpose, we use a multi-layer long short-term memory (LSTM) network and consider a sliding window of size w for encoding. Define $\mathbf{X}_{t:(t+w-1)} := \{X_{t:(t+w-1)}^1; \ldots; X_{t:(t+w-1)}^d\}$ that consists of all time series in the sliding window. An LSTM network f_θ maps $\mathbf{X}_{t:(t+w-1)} \in \mathbb{R}^{w \times d}$ to $H_{enc} \in \mathbb{R}^{h_1 \times d}$ hidden features where θ indicates the parameter of the model, i.e.,

$$H_{enc} = f_\theta(\mathbf{X}_{t:(t+w-1)}). \quad (1)$$

For each sliding window, we reset all hidden state values to zero before the computation of encoding. The values of the last hidden layer H_{enc} are passed to the next module for the aggregation of messages.

Aggregation. The idea of this module is to use the adjacency matrix A to represent a weighted summary graph where elements in column j represent the parental contributions of all time series to time series j. Then, all time series are weighted by A and aggregated using a graph neural network. To normalize the weights, we first define a matrix $\tilde{A} \in \mathbb{R}^{d\times d}$ as trainable parameters that are randomly initialized. It is then passed through a sigmoid function σ to convert each of the weights to a probability that represents the normalized weights of each edge in the causal graph, i.e., $A = \sigma(\tilde{A})$.

One challenge of this module is that the model may not be sensitive to the accuracy of the adjacency matrix and it may tend to aggregate the information from all time series. To improve the performance of the model, we consider two parts in message aggregation. The first part is a one-layered graph convolutional network (GCN) that provides an aggregation of the encoded information from the input matrix H_{enc} directly according to the parents of each time series given in A. The aggregated features are then passed through a linear transformation, as shown in Eq. (2) where $W_1 \in \mathbb{R}^{h_1\times h_3}$ is the parameter matrix. The second part is a two-layer GCN as shown in Eq. (3), where $W_2 \in \mathbb{R}^{h_1\times h_2}$ and $W_3 \in \mathbb{R}^{h_2\times h_3}$ are weight matrices of the stacked layers and ReLU is the activation function. The two parts are balanced by a parameter α before summing together to adjust the effect of each term as required. In this way, the aggregation layer accumulates the messages from all variables and passes them to the decoder module. The reason we consider the second part is that it represents how the predicted features of the parents can be used to predict the future values of a time series. It increases the sensitivity of the model to the accuracy of the adjacency matrix A to the true causal graph since the more accurate the matrix A is, the more useful predicted features of the parents can be obtained. In the experiments, we observed that $\alpha = 0.9$ achieved the best performance.

$$H_{agg1} = AH_{enc}W_1 \tag{2}$$

$$H_{agg2} = A\mathit{ReLU}(AH_{enc}W_2)W_3) \tag{3}$$

$$H_{agg} = \alpha \cdot H_{agg1} + (1-\alpha)\cdot H_{agg2} \tag{4}$$

Decoding. This module is a multilayer perceptron (MLP) network with one fully connected layer followed by a sigmoid layer to output a normalized prediction. The MLP decodes the aggregated message from the previous module into the predicted values of the time series at the next time step, i.e.,

$$\hat{X}_{t+w} = g_\phi(H_{agg}). \tag{5}$$

D. Loss Function and Thresholding

To learn the causal structure A, we define the score function as a regularized loss as shown in Eq. (6):

$$L = \frac{1}{n-w}\sum_{t=0}^{n-w-1}(X_{t+w}-\hat{X}_{t+w})^2+\lambda(||A||_2+(A-\bar{A})^2), \tag{6}$$

where $\bar{A}$ represents the column-wise mean of A. The loss function consists of three parts. The first part is the mean squared error that measures the accuracy of the reconstruction of the time series data. The second part is the L2 norm that controls the magnitude of matrix A. The last part is a regularization term that aims to reduce the difference between each value in A and the mean of the column to which this value belongs. The purpose is to use the mean of each column as the threshold for determining the edges in the causal graph. Finally, we convert A to a summary causal graph such that there is an edge pointing from time series i to time series j if $A_{i,j}$ is greater than the mean of column j.

E. Training

The entire model is continuously optimized by gradient descent to minimize the above loss function. The data are first normalized to fall in the range [0-1] before the training. Then, as mentioned above, a sliding-window technique is used to preprocess the input data. We use a sliding window of length w to divide each time series into $(n-w)$ slices, moving the window by 1 step each time. As a result, the training data is given by $\mathcal{X} = \{\mathbf{X}_{0:(w-1)}, \mathbf{X}_{1:w}, ..., \mathbf{X}_{(n-w-1):(n-2)}\}$. For each window, the model predicts the values of the next time step in the time series. Thus, the target value for prediction is defined as X_t for each window $\mathbf{X}_{(t-w):(t-1)}$, such that $\mathcal{Y} = \{X_w, X_{w+1}, ..., X_{n-1}\}$. In the experiments, we use the window size w = 5. The preprocessed $\mathcal{X}$ is fed to our model batch-wise in a sequential manner, one variable after another. During the training, the adjacency matrix A is updated such that the weights of the correct parents are increased and the prediction error is minimized. The training stops when the loss converges.

IV. Experiments

A. Experimental Setting

In the experiments, we perform gradient descent using the full dataset for every training iteration. For the GCN layer, we use $h_1 = h_2 = h_3$, with $\alpha = 0.9$. We train the model to learn the adjacency matrix; the normalized matrix is converted to a binary matrix using the column-wise mean as the threshold. We evaluate our model on several synthetic data and a real-world dataset. All the experiments are implemented in PyTorch and run on a Ubuntu 20.04.4 LTS computer with Intel(R) Core(TM) i9-10900X CPU and NVIDIA GeForce RTX 3080 10GB GPU. All the data and code are available at https://anonymous.4open.science/r/NTiCD/.

B. Baselines

We compare NTiCD with five baselines, VarLinGAM [13] TCDF [20], DYNOTEARS [22], GVAR [19], and FCITMI [2]. The baselines were chosen to cover the categories of Granger causality-based, constraint-based, and score-based techniques.

VarLiNGAM is Granger causal discovery approach that combines the non-Gaussian instantaneous model with autoregressive models to estimate a structural vector autoregression

(SVAR) model. It is a temporal extension of LiNGAM, where the model is estimated through a least-square procedure.

Temporal Causal Discovery Framework (TCDF) is a deep learning-based framework that learns both window and summary causal graphs using an attention-based convolutional neural network. Their model has four steps: Time Series Prediction, Attention Interpretation, Causal Validation, and Delay Discovery. It uses different independent attention-based CNNs, with the same architecture, to predict the different time series in the data. We have used the hyperparameter values as suggested by the authors to run our synthetic data: a kernel of size 4, a dilation coefficient equal to 4, 1 hidden layer, a learning rate of 0.01, and 5000 epochs.

DYNOTEARS is a method for learning dynamic Bayesian networks that can simultaneously estimate both contemporaneous (intra-slice) and time-lagged (inter-slice) relationships between the variables in time series data. It is a score-based technique that optimizes by minimizing a loss, based on the acyclicity constraint. We have set the hyperparameters to their recommended values for all the experiments.

Generalized vector autoregression (GVAR) is also a score-based framework for learning non-linear multivariate Granger causality using an extension of self-explaining neural networks (SENNs). The uniqueness of this method is that, apart from inferring the causal relations, GVAR can also detect signs of Granger-causal effects and inspect their variability over time. The authors used K-layered multilayer perceptrons (MLPs) to represent the non-linear function of the vector autoregression for lag K. They train two different GVAR models with original and time-reversed data for more stable structure identification in order to deduce the final output binary relationship. We used all the hyperparameter values recommended by the authors for their experiments on linear VAR: 2 hidden layers, each of size 50, 64 batch sizes, 0.0001 learning rate, and 1000 training epochs.

FCITMI is a constraint-based method that learns summary causal graphs on time series by combining the two famous algorithms PC and FCI. They use a new mutual information measure determined by a sliding window technique for the conditional independence test. However, they assume first-order self-causation of the time series, unlike our method. The result of FCITMI is a PAG that outputs a matrix that includes an undetermined relationship. Thus, we ignore all undirected edges of their output graph to convert their result to a summary causal graph comparable with our method.

Since VarLiNGAM, DYNOTEARS, and TCDF output window causal graphs, we convert them to a summary graph by considering an edge from one variable to another if there's an edge with any lag between the corresponding variables in the window causal graph. The rest of the methods provides a summary causal graph that can be directly compared with our method.

C. Datasets

1) Synthetic Data: We generate synthetic data based on two settings: Syn-1 is the scalar-valued time series dataset where each of the variables in the time series is one-dimensional; Syn-2 is the multidimensional time series dataset, as discussed below. This aims to assess NTiCD's performance in two areas: first, how it compares with the state-of-the-art approaches for inferring the summary causal graph; second, how it measures up to itself for scalar and vector-valued variables. The subsequent discussion includes a thorough explanation of data production and experimental findings.

Syn-1. In this setting, we construct multivariate data with scalar values using the approach outlined below:

$$X_t = A^T \sum_{j=1}^{5} \beta_j \cos(X_{t-j} + 1) + \epsilon \tag{7}$$

where X_t represents a vector of d variables at time step t, β is the regression coefficient, and ϵ represents standard Gaussian noise. The noise scale is less than 1 and is proportional to the value of d. The $\cos$ function is used to create a non-linear relationship between time series. We use a window size of 5 and define the initial values $\mathbf{X}_{0:4}$ at random. We generate multiple datasets with different underlying causal graphs and different numbers of nodes by selecting different ground truth adjacency matrices A randomly.

Syn-2. In this setting, we generate multisequence time series data in a way similar to Syn-1, with the exception that the variables have multiple features as shown in Eq. (8).

$$X_t[s] = A^T \sum_{j=1}^{5} \beta_j^s \cos(X_{t-j}[s] + 1) + \epsilon[s] \tag{8}$$

where $X_t[s]$ represents the sth feature at time-step t, which is a vector of size d. The underlying causal structure of all s-sequences is the same, with only the regression coefficients and noise being different (as indicated by $\epsilon[s]$). With this configuration, we examine the vector-valued case of NTiCD. This means that the dimension of each of our data sets is $n \times d \times s$. In our experiments, we use the $s = 5$ features. Our model architecture is able to handle multi-dimensional data and all the preprocessing and computations are done in a similar way to Syn-1.

2) Real Data: To see how NTiCD performs in a real-world setting, we apply our model to the Netsim dataset [27]. It is made up of realistic simulated functional magnetic resonance imaging (fMRI) time series data that represents the blood-oxygen-level-dependent (BOLD) across several regions of the human brain. The underlying connectivity in this case demonstrates the causal relationships between the various brain regions, hence the adjacency matrix here depicts the connection relationship and the nodes represent various brain regions. The dataset consists of simulations of a large number of different brain areas and underlying causal matrices, from which we use the third simulation Sim-3.mat. This data set includes samples from 15 different brain areas, each of which has a length of 200 for 50 participants. We adopted this simulation because the underlying causal graph is the same for all of the subjects.

TABLE I: Hyperparameter values used for training NTiCD with different datasets.

Experiment	d	h_1	# Layers	LR	Batch-size	Epochs
Syn-1	6	128	5	10^{-5}	128	1500
	10, 20, 50	128	5	10^{-5}	128	3000
Syn-2	6, 10	128	5	10^{-5}	128	10000
Netsim data	15	50	2	10^{-4}	16	10000

d = no. of variables, h_1 = size of LSTM layer

TABLE II: The performance of NTiCD for multisequence data of Syn-2.

Data	Accuracy	Precision	Recall	F1-score
6-var	0.81 ± 0.06	0.87 ± 0.11	0.64 ± 0.02	0.73 ± 0.12
10-var	0.86 ± 0.4	0.94 ± 0.03	0.64 ± 0.08	0.76 ± 0.07

TABLE III: Evaluating the scalability of NTiCD, TCDF, and DYNOTEARS with single sequence data.

Method	d	Accuracy	Precision	Recall	F1-score
TCDF	6	0.69 ± 0.032	0.61 ± 0.15	0.39 ± 0.07	0.48 ± 0.09
	10	0.56 ± 0.03	0.53 ± 0.40	0.12 ± 0.07	0.16 ± 0.07
	20	0.52 ± 0.01	0.64 ± 0.08	0.12 ± 0.13	0.07 ± 0.04
	50	0.50 ± 0.01	0.39 ± 0.11	0.009 ± 0.01	0.02 ± 0.01
DYNOTEARS	6	0.49 ± 0.04	0.42 ± 0.02	0.94 ± 0.10	0.58 ± 0.03
	10	0.43 ± 0.11	0.41 ± 0.09	0.74 ± 0.43	0.46 ± 0.11
	20	0.54 ± 0.03	0.57 ± 0.06	0.28 ± 0.12	0.36 ± 0.09
	50	0.05 ± 0.01	0.5 ± 0.06	0.16 ± 0.12	0.23 ± 0.25
NTiCD	6	0.80 ± 0.04	0.71 ± 0.07	0.78 ± 0.12	0.74 ± 0.06
	10	0.69 ± 0.12	0.64 ± 0.09	0.65 ± 0.14	0.64 ± 0.11
	20	0.56 ± 0.02	0.55 ± 0.04	0.50 ± 0.02	0.52 ± 0.03
	50	0.50 ± 0.01	0.50 ± 0.01	0.37 ± 0.03	0.43 ± 0.02

D. Results

We evaluate the performance of our model in terms of four metrics: accuracy, precision, recall, and F1-score. The hyperparameters used for training NTiCD for each of the datasets discussed in the above subsection are provided in Columns 3-7 of Table I, where d is the number of variables, h_1 is the size of each layer, # Layers indicates the number of layers of LSTM, and LR denotes learning rate. We used a smaller dimension of the hidden layer and mini-batch size for the Netsim dataset since the length of each data is only 200.

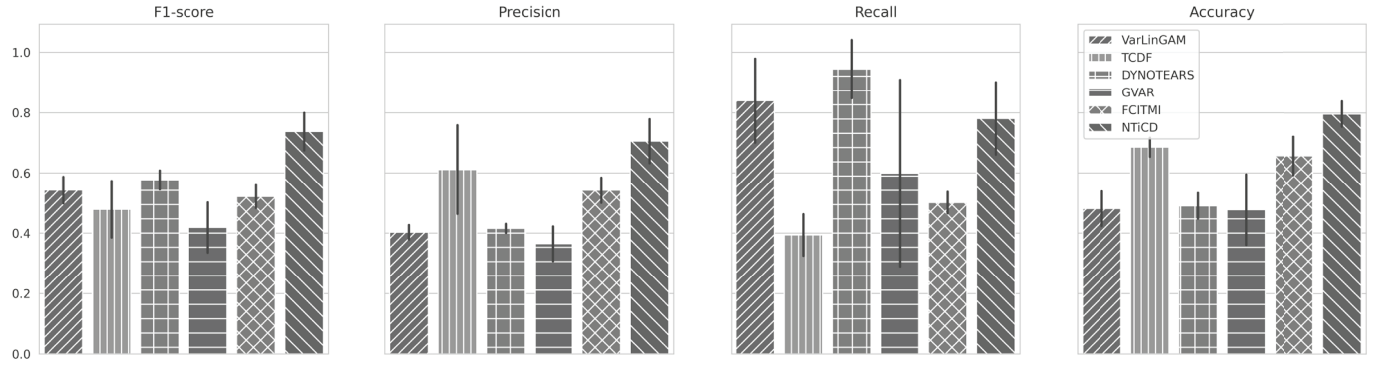
1) Evaluating Causal Discovery Performance: First, we perform experiments with single sequence data according to Syn-1 to evaluate the performance of NTiCD in discovering the summary causal graph. The results compared with the five baselines for synthetic data with 6 and 10 variables are shown in Fig. 3. We show the mean and standard deviation of 3 simulations for each number of variables in our experiments where we trained the model separately on 3 different causal graphs for each variable number. As can be seen, NTiCD produces the best F1-score, precision, and accuracy among all methods in both the 6-variable and 10-variable settings. VarLiNGAM and DYNOTEARS have high recall but low precision, implying that they tend to predict more edges in the graph. On the other hand, TCDF produces the second highest precision but extremely low recall, implying that it tends to predict a sparse graph.

Next, we inspect the behavior of NTiCD for multisequence data as generated in Syn-2. We provide the results of 6 and 10-variable multisequence data in Table II. Since all the baselines are not compatible with multi-dimensional data, we only show the results of our method. Similarly, we compute the mean and standard deviation of the results across 3 simulations for three different causal graphs. The results show that NTiCD is well-suited for multidimensional variables and performs better compared to single sequence data, implying that NTiCD is capable of leveraging the advantage of multisequence data.

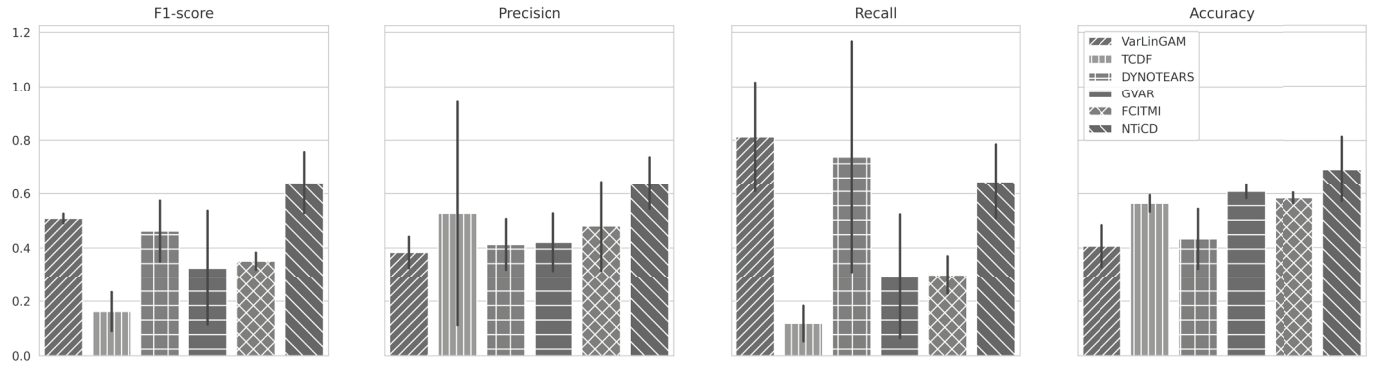
We apply our model to the Netsim dataset to assess its performance in real world setting. The results are provided in Fig. 4, where we show the mean values of the simulation of five data (subjects 1-5). We also provide the performance results of the five baseline methods. We have used the hyperparameter values for the baselines as suggested by the authors in their studies. For VarLinGAM, we use $\alpha = 0.01$ and $\tau = 1$. For DYNOTEARS, we set the τ_w value to 0.3, and use other hyperparameters values as before. We changed the kernel size to 1, the learning rate to 10^{-3}, and used 2000 epochs for TCDF. In the case of GVAR, the batch-size is changed to 256 and the number of hidden layers to 1, while the rest of the hyperparameters are the same as discussed in Section IV-B. We used a slightly smaller LSTM (as provided in Table I) for this dataset as it has a smaller number of time steps (200), thus a complex architecture would overfit the data resulting in worse performance. As we see from the results in Fig. 4, the best-performing model in terms of F1-score and accuracy is VarLinGAM, indicating that the fMRI data follow a linear pattern [2]. However, NTiCD still achieves comparable performance to VarLinGAM and produces the highest recall among all methods.

2) Evaluating Scalability: In order to evaluate the scalability of NTiCD, we generate 20 and 50 variable synthetic data following Syn-1. We compare the results of NTiCD with those of DYNOTEARS and TCDF as other baselines do not scale well to large datasets. The summary of the results is provided in Table III. From these simulations, we can see that the performance of NTiCD does not degrade too much when the number of variables increases. It outperforms DYNOTEARS and TCDF in most cases in terms of all metrics. On the other hand, the performance of TCDF degrades significantly when the number of variables is 50, producing almost zero recall.

3) Evaluating Impact of α: Finally, we inspect the significance of varying the parameter α, as defined in Eq. (4), on the overall performance of NTiCD for causal graph discovery. We use synthetic data with six variables for this purpose. We run several experiments with varying α on three causal graphs, and show the variation of mean performance for a range of α in Fig. 5. As seen in the graph, the peak performance is obtained at $\alpha = 0.9$, which has been used for simulation in all our experiments.



(a) 6-variable



(b) 10-variable

Fig. 3: Causal discovery performance of different methods on Syn-1 data with both 6-variable and 10-variable settings in terms of F1-score, precision, recall, and accuracy.

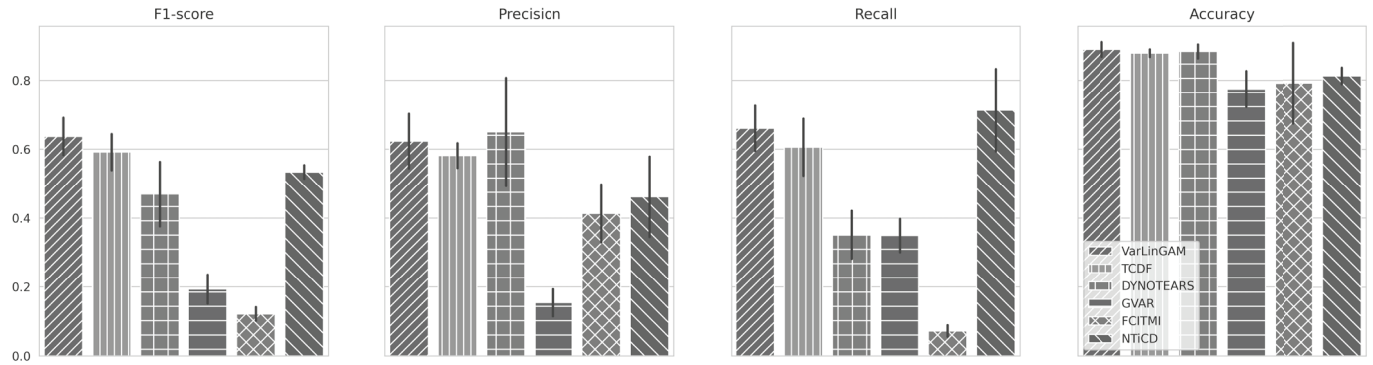


Fig. 4: Causal discovery performance of different methods on Netsim dataset in terms of F1-score, precision, recall, and accuracy.

V. Conclusion

In this paper, we proposed a graph neural network-based causal discovery algorithm, referred to as NTiCD. It is a score-based method that learns a summary causal graph from multivariate temporal data using the principle of Granger Causality. It applies an encoder-decoder model where we use an LSTM as the encoder to learn the complex hidden features of the variables in the observational time series. These features are then passed to a graph convolutional network (GCN) layer to aggregate the message from all nodes to pass to a simple MLP that works as the decoder. Thus, the model uses time series prediction to discover the causal structure in the data. We conducted several experiments with synthetic and real-world datasets to evaluate the performance of our model. NTiCD is applicable to both single and multi-sequence data, as well as to both linear and non-linear data. We proved the performance of the proposed method for various datasets by comparing it with five other state-of-the-art baseline methods. Since NTiCD does not presume acyclicity, it can also identify self-loops in the data. One limitation of the model is data

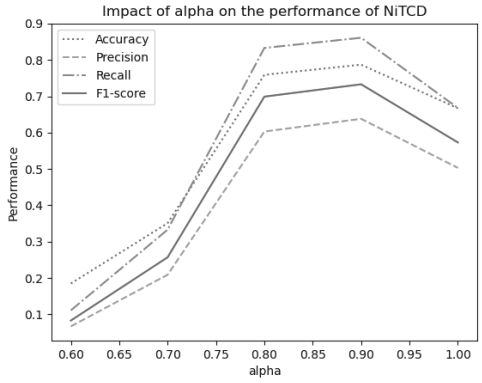


Fig. 5: The change in causal discovery performance of NTiCD with the variation of α in terms of accuracy, precision, recall, and F1-score.

scaling: NiTCD does not perform as well as it does for small datasets. In future work, we will improve the data efficiency of the model by learning hidden representations for exogenous variables.

REFERENCES

[1] Saima Absar and Lu Zhang. Discovering time-invariant causal structure from temporal data. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2807–2811, 2021.

[2] Charles K Assaad, Emilie Devijver, and Eric Gaussier. Entropy-based discovery of summary causal graphs in time series. *Entropy*, 24(8):1156, 2022.

[3] Charles K Assaad, Emilie Devijver, and Eric Gaussier. Survey and evaluation of causal discovery methods for time series. *Journal of Artificial Intelligence Research*, 73:767–819, 2022.

[4] Sahar Behzadi, Benjamin Schelling, and Claudia Plant. Itgh: Information-theoretic granger causal inference on heterogeneous data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 742–755. Springer, 2020.

[5] Doris Entner and Patrik O Hoyer. On causal discovery from time series data using fci. *Probabilistic graphical models*, pages 121–128, 2010.

[6] Yinghua Gao, Li Shen, and Shu-Tao Xia. Dag-gan: Causal structure learning with generative adversarial nets. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3320–3324. IEEE, 2021.

[7] Andreas Gerhardus and Jakob Runge. High-recall causal discovery for autocorrelated time series with latent confounders. *Advances in Neural Information Processing Systems*, 33:12615–12625, 2020.

[8] Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 10:524, 2019.

[9] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[10] Clive W.J. Granger. Time series analysis, cointegration, and applications. *American Economic Review*, 94(3):421–425, June 2004.

[11] Ruocheng Guo, Lu Cheng, Jundong Li, P Richard Hahn, and Huan Liu. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys (CSUR)*, 53(4):1–37, 2020.

[12] Biwei Huang, Kun Zhang, Yizhu Lin, Bernhard Schölkopf, and Clark Glymour. Generalized score functions for causal discovery. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1551–1560, 2018.

[13] Aapo Hyvärinen, Kun Zhang, Shohei Shimizu, and Patrik O Hoyer. Estimation of a structural vector autoregression model using non-gaussianity. *Journal of Machine Learning Research*, 11(5), 2010.

[14] Christopher Krich, Jakob Runge, Diego G Miralles, Mirco Migliavacca, Oscar Perez-Priego, Tarek El-Madany, Arnaud Carrara, and Miguel D Mahecha. Estimating causal networks in biosphere–atmosphere interaction with the pcmci approach. *Biogeosciences*, 17(4):1033–1061, 2020.

[15] Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural dag learning. *arXiv preprint arXiv:1906.02226*, 2019.

[16] Yan Liu, Alexandru Niculescu-Mizil, Aurelie C Lozano, and Yong Lu. Learning temporal causal graphs for relational time-series analysis. In *ICML*, 2010.

[17] Sindy Löwe, David Madras, Richard Zemel, and Max Welling. Amortized causal discovery: Learning to infer causal graphs from time-series data. In *Conference on Causal Learning and Reasoning*, pages 509–525. PMLR, 2022.

[18] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[19] Ričards Marcinkevičs and Julia E Vogt. Interpretable models for granger causality using self-explaining neural networks. *arXiv preprint arXiv:2101.07600*, 2021.

[20] Meike Nauta, Doina Bucur, and Christin Seifert. Causal discovery with attention-based convolutional neural networks. *Machine Learning and Knowledge Extraction*, 1(1):19, 2019.

[21] Ignavier Ng, Shengyu Zhu, Zhitang Chen, and Zhuangyan Fang. A graph autoencoder approach to causal structure learning. *arXiv preprint arXiv:1911.07420*, 2019.

[22] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*, pages 1595–1605. PMLR, 2020.

[23] Jakob Runge. Causal network reconstruction from time series: From theoretical assumptions to practical estimation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(7):075310, 2018.

[24] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science advances*, 5(11):eaau4996, 2019.

[25] Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

[26] Ali Shojaie and Emily B Fox. Granger causality: A review and recent advances. *Annual Review of Statistics and Its Application*, 9:289–319, 2022.

[27] Stephen M Smith, Karla L Miller, Gholamreza Salimi-Khorshidi, Matthew Webster, Christian F Beckmann, Thomas E Nichols, Joseph D Ramsey, and Mark W Woolrich. Network modelling methods for fmri. *Neuroimage*, 54(2):875–891, 2011.

[28] Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social science computer review*, 9(1):62–72, 1991.

[29] George Sugihara, Robert May, Hao Ye, Chih-hao Hsieh, Ethan Deyle, Michael Fogarty, and Stephan Munch. Detecting causality in complex ecosystems. *science*, 338(6106):496–500, 2012.

[30] Xiaohai Sun. Assessing nonlinear granger causality from multivariate time series. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 440–455. Springer, 2008.

[31] Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily B Fox. Neural granger causality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(8):4267–4279, 2021.

[32] Matthew J Vowels, Necati Cihan Camgoz, and Richard Bowden. D'ya like dags? a survey on structure learning and causal discovery. *ACM Computing Surveys*, 55(4):1–36, 2022.

[33] Tailin Wu, Thomas Breuel, Michael Skuhersky, and Jan Kautz. Discovering nonlinear relations with minimum predictive information regularization. *arXiv preprint arXiv:2001.01885*, 2020.

[34] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. Dag-gnn: Dag structure learning with graph neural networks. In *International Conference on Machine Learning*, pages 7154–7163. PMLR, 2019.

[35] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[36] Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric Xing. Learning sparse nonparametric dags. In *International Conference on Artificial Intelligence and Statistics*, pages 3414–3425. PMLR, 2020.